references. The Applicants noted that the Copeland and Colwell references even in combination do not teach or suggest all of the recited elements in each of the independent claims. Although no agreement was reached, the Examiner noted that he would reconsider the rejection based on Copeland and Colwell.

Claims 1, 9, 16, and 23 variably recite an interface "allowing the use of new command objects without modifying the application."

Copeland describes a system for retrieving data that is arbitrarily organized. "Computer systems organize data sources in a variety of ways. For example, data can be organized as a relational database, as a hierarchical database, as a flat file, as a spreadsheet, etc. Application programs, in general, can only access data sources with organizations and implementations that the developers of the applications program were aware of at the time of development. Thus, these applications not only cannot access other data sources, but also may not be able to access revisions to these data sources" (column 1, lines 12-22). Copeland provides a uniform data interface (UDI) system that allows "for uniformly interfacing with data from a data source that is arbitrarily structured" (column 1, lines 30-32). "Application programs are developed to access data sources using the UDI API. Thus, the application program can access any data source that has a UDI mapper to map its organization to the UDI model." However, Copeland does not teach or suggest an interface "allowing the use of new command objects without modifying the application." Copeland merely provides that an application can access arbitrary data. In fact, it is believed that Copeland does not even teach or suggest command objects.

Colwell describes a system for searching and retrieving text stored in files. In response to a word search request initiated by a user, the system builds index files representing the approximate position and frequency of every word in the search request within files on a given storage unit (column 2, lines 18-20). A viewer module is invoked to view the file containing the words in the search request. Colwell does not teach or suggest an interface "allowing the use of new command objects without modifying the application." By contrast, Colwell describes a user manually invoking a software application by loading the application corresponding to the viewer (Column 2, lines 33-36) or the application itself loading an appropriate viewer (Column 2, lines 36-39). Applicants understand that the Colwell description requires that the application itself be modified in order to use a new command object such as a new viewer. Applicants are unaware of any description in Colwell that teaches or suggests an interface that allows the use of

command objects without modifying the application. According to claims 9, 16, and 25, the interface is "independent from the application" and allows the use of new command objects without modifying the application. According to claim 1, the interface is a "generic interface" that allows the use of new command objects without modifying the application.

Furthermore, independent claims 1 and 23 explicitly recite "a data handler mechanism arranged to interface with an application." To more clearly describe a data handler mechanism, examples will be used for clarity. However, it should be noted that the claim recitation should in no way be limited to the example described. In one example, "a data handler is arranged to access data in response to a request from the application" (page 9, lines 1-2). A command map can also be used by a data handler "to locate an appropriate command object" (page 10, line 5-6). The Examiner argued that Colwell describes this element. However, it is believed that neither Copeland nor Colwell teach or suggest a data handler mechanism. As noted during the telephone conference, the user interface described in Colwell is not a data handler mechanism. In addition, claims 1 and 23 recite that the data handler mechanism is in communication with the data retriever mechanism. Colwell neither discloses nor suggests a data handler mechanism "arranged to interface with an application" where the data handler mechanism is "in communication with a data retriever mechanism."

In view of the remarks noted above, the Applicants believe the rejections to base claims 1, 9, 16, and 23 have been traversed thereby placing claims 2-8, 10-15, 17-22, and 24-27 in condition for allowance in their present form for at least the reasons noted above.

In view of the above, Applicants believe that all pending claims are allowable and respectfully request a Notice of Allowance for this application from the Examiner. If the Examiner has any questions or concerns, please feel free to contact the undersigned at the telephone number set out below.

If any fees are due in connection with the filing of this amendment, the Commissioner is authorized to charge such fees to Deposit Account 50-0388 (Order No. SUN1P123). A duplicate copy of the transmittal is enclosed for this purpose.

Respectfully submitted,
BEYER WEAVER & THOMAS, LLP

Godfrey K. Kwan
Reg. No. 46,850

P.O. Box 778
Berkeley, California 94704-0778
(510) 843-6200

APPENDIX

1. (Amended) A computer-implemented framework for associating data with a command object, the command object being arranged to operate on the data, wherein the data is associated with an application, the computer-implemented framework comprising:

a data handler mechanism arranged as a generic interface with the application, wherein the generic interface allows use of new command objects without modifying the application;

a data retriever mechanism in communication with the data handler mechanism, the data retriever mechanism being arranged to obtain the data and to pass the data to the data handler mechanism; and

a mapping mechanism in communication with the data handler mechanism, the mapping mechanism being separate from the data handler mechanism, the mapping mechanism being arranged to obtain the command object, wherein the command object is obtained by the mapping mechanism based substantially on the data.

2. A computer-implemented framework according to claim 1 wherein the data is a stream of bytes, and the data handler mechanism is further arranged to bind the stream of bytes to the command object.

3. A computer-implemented framework according to claim 1 wherein the data retriever mechanism includes a data content handler mechanism in communication with the data handler mechanism, the data content handler mechanism being arranged to convert the data into a data object, wherein the data handler mechanism is further arranged to bind the data object to the command object.

4. A computer-implemented framework as recited in claim 3 wherein the data object is created using the Java™ programming language, and the command object is a Java™ command object.

5. A computer-implemented framework as recited in claim 1 wherein the data is one of text data and image data.

6. A computer-implemented framework as recited in claim 1 wherein the data handler is further arranged to receive a request from the application, to bind the data to the command object, and to return the command object to the application.

7. A computer-implemented framework as recited in claim 1 wherein the data retriever includes a data source mechanism arranged to obtain a stream of bytes and a data content handler mechanism arranged to convert the stream of bytes into a data object, the data source mechanism being in communication with the data content handler mechanism, wherein the data handler mechanism is further arranged to bind the data object to the command object.

8. A computer-implemented framework as recited in claim 1 wherein the mapping mechanism includes a look-up table arranged to associate the command object with the data.

9. (Amended) A computer-implemented method for associating data with a command object in response to a request from an application, the method comprising:

accessing the data through an interface in response to the request from the application, the interface being independent from the application and in communication with the application, wherein the request from the application is processed by the interface, the interface allowing use of new command objects without modifying the application;

accessing a mapping mechanism which is in communication with the interface, the mapping mechanism being independent from the application such that the mapping mechanism is not a component of the application, the mapping mechanism being maintained separately from the interface, the mapping mechanism further being arranged to locate a command object that is appropriate for the data, wherein the mapping mechanism is accessed by the interface;

obtaining the command object that is appropriate for the data, wherein the mapping mechanism obtains the command object and passes the obtained command object to the interface;

binding the command object to the data, wherein the interface binds the command object to the data; and

returning the command object to the application, wherein the interface returns the command object to the application.

10. A computer-implemented method as recited in claim 9 wherein accessing the data through an interface includes:

passing a stream of bytes to a data content handler mechanism arranged to create a data object from the stream of bytes; and

passing the data object to the interface, wherein the data is the data object.

11. A computer-implemented method as recited in claim 10 wherein the data object is created using the Java™ programming language, and the command object is a Java™ command object.

12. A computer-implemented method as recited in claim 9 wherein accessing the data through the interface includes accessing a data retriever which is arranged to obtain the data, wherein the data is a stream of bytes.

13. A computer-implemented method as recited in claim 9 further including operating on the data using the command object.

14. A computer-implemented method as recited in claim 9 wherein the command object that is appropriate for the data is selected from a set of command objects associated with a command list, the command list being associated with the data, the method further including accessing the command list, wherein the command list is accessed by the interface.

15. A computer-implemented method as recited in claim 14 wherein accessing the command list includes receiving a request for a command list from the application, the request for the command list being received by the interface, wherein the interface performs the steps of:

obtaining a type associated with the data;

obtaining the command list through the mapping; and

returning the command list to the application.

16. (Amended) A computer program product for associating data with a command object in response to a request from an application, the computer program product comprising:

computer code for accessing the data through an interface in response to the request from the application, the interface being independent from the application and in communication with the application, wherein the request from the application is processed by the interface, the interface allowing use of new command objects without modifying the application;

computer code for accessing a mapping mechanism which is in communication with the interface, the mapping mechanism being independent from the application such that the mapping mechanism is not a part of the application, the mapping mechanism further being separately maintained from the interface, the mapping mechanism further being arranged to locate a command object that is appropriate for the data, wherein the mapping mechanism is accessed by the interface;

computer code for obtaining the command object that is appropriate for the data, wherein the mapping mechanism obtains the command object and passes the obtained command object to the interface;

computer code for binding the command object to the data, wherein the interface binds the command object to the data;

computer code for returning the command object to the application, wherein the interface returns the command object to the application; and

a computer-readable medium that stores the computer codes.

17. A computer-readable medium as recited in claim 16 wherein the computer program code devices configured to cause the computer to access the data through an interface include computer program code devices configured to cause a computer to execute the steps of:

passing a stream of bytes to a data content handler mechanism arranged to create a data object from the stream of bytes; and

passing the data object to the interface, wherein the data is the data object.

18. A computer-readable medium as recited in claim 17 wherein the data object is created using the Java™ programming language, and the command object is a Java™ command object.

19. A computer-readable medium as recited in claim 16 further including computer program code devices configured to cause the computer to operate on the data using the command object.

20. A computer-readable medium as recited in claim 16 wherein the command object that is appropriate for the data is selected from a set of command objects associated with a command list, the command list being associated with the data, the computer-readable medium further including computer code devices configured to cause the computer to access the command list through the interface.

21. A computer-implemented framework according to claim 1 wherein the command object is obtained by the mapping mechanism based substantially on the data without an external input from a user of the application.

22. A computer-implemented framework according to claim 1 wherein the command object is obtained by the mapping mechanism based substantially on the data without directly involving the application.

23. (Amended) A computer-implemented framework for associating data with a command object, the command object being arranged to operate on the data, wherein the data is associated with a selected application, the computer-implemented framework comprising:

a data handler mechanism arranged to interface with a plurality of applications, the plurality of applications including the selected application, wherein the data handler mechanism is independent from the plurality of applications and allows use of new command objects without modifying the application;

a data retriever mechanism in communication with the data handler mechanism, the data retriever mechanism being arranged to obtain the data and to pass the data to the data handler mechanism; and

a mapping mechanism in communication with the data handler mechanism, the mapping mechanism being substantially separate from the data handler mechanism, the mapping mechanism being arranged to obtain the command object, wherein the mapping mechanism is associated with the plurality of applications and is arranged to obtain the command object without directly involving the selected application.

24. A computer-implemented framework as recited in claim 23 wherein the mapping mechanism and the data handler mechanism are separately maintained.

25. A computer-implemented framework as recited in claim 23 wherein the mapping mechanism is not a component of the data handler mechanism.

26. A computer-implemented framework as recited in claim 1 wherein the mapping mechanism and the data handler mechanism are separately maintained.

27. A computer-implemented framework as recited in claim 1 wherein the mapping mechanism is not specific to the application while the data handler mechanism is substantially specific to the application.